106061151 劉安得 HW4

*1. (30%)*

*MCU8051IDE中有提供一個4 x 4 keypad以及1個數字顯示的7段顯示器的Virtual Hardware，請設計一個程式來讀入按鍵及顯示，一開始沒有任何按鍵時先不顯示7段顯示器，當開始有按鍵時，把該按鍵對應的文數字顯示在7段顯示器上，之後的按鍵就依此類推。你最少必須提供流程圖(或pseudo code)，解釋你的做法，你的程式碼，Virtual Hardware儲存檔。可以讓助教驗證你的程式。*

MAIN()

BEGIN

WHILE [1] DO BEGIN

[call subroutine inhex()]

[call subroutine outchr()]

END

END

INHEX()

BEGIN

[set counter = 50]

WHILE [counter > 0] DO BEGIN

[call subroutine get\_key()]

IF [key pressed] THEN

[counter --]

ELSE

[set counter = 50]

END

[set counter = 50]

WHILE [counter > 0] DO BEGIN

[call subroutine get\_key()]

IF [key release] THEN

[counter --]

ELSE

[set counter = 50]

END

RETURN()

END

GET\_KEY()

BEGIN

[set column counter = 4]

[start with column 0]

WHILE [column counter > 0] DO BEGIN

[activate column line]

[read port 0]

IF [row line active] THEN BEGIN

[column number = 4 – column counter]

[get row number]

[hex code = column number + 4\* row number]

[set c = 1]

RETURN()

END

ELSE BEGIN

[move to next column line]

[column counter --]

END

END

[clear c]

RETURN()

END

OUTCHR()

BEGIN

CASE [A] OF

00H: BEGIN

[P2 = 03H]

RETURN()

END

01H BEGIN

[P2 = 9FH]

RETURN()

END

02H: BEGIN

[P2 = 25H]

RETURN()

END

03H: BEGIN

[P2 = 0DH]

RETURN()

END

04H: BEGIN

[P2 = 99H]

RETURN()

END

05H: BEGIN

[P2 = 49H]

RETURN()

END

06H: BEGIN

[P2 = 41H]

RETURN()

END

07H: BEGIN

[P2 = 1FH]

RETURN()

END

08H: BEGIN

[P2 = 01H]

RETURN()

END

09H: BEGIN

[P2 = 09H]

RETURN()

END

0AH: BEGIN

[P2 = 11H]

RETURN()

END:

0BH: BEGIN

[P2 = 0C1H]

RETURN()

END

0CH: BEGIN

[P2 = 63H]

RETURN()

END

0DH: BEGIN

[P2 = 85H]

RETURN()

END:

0EH: BEGIN

[P2 = 61H]

RETURN()

END

0FH: BEGIN

[P2 = 71H]

RETURN()

END

ENDCASE

END

MAIN: CALL IN\_HEX ;get hex code from keypad

CALL OUTCHR

SJMP MAIN ;repeat

IN\_HEX: MOV R3, #50 ;debouncecount

BACK: CALL GET\_KEY ;key pressed? C=1 yes, C=0 no

JNC IN\_HEX ;no: check again

DJNZ R3, BACK ;yes: repeat 50 times for debouncing

PUSH ACC ;save hex code

BACK2: MOV R3, #50 ;wait for key up

BACK3: CALL GET\_KEY ;key still pressed?

JC BACK2 ;yes: keep checking

DJNZ R3, BACK3 ;no: key release, repeat 50 times

POP ACC ;recover hex code and

RET ;return

GET\_KEY: MOV A, #0FEH ;start with column 0

MOV R6, #4 ;use R6 as column counter

TEST: MOV P1, A ;activate column line

MOV R7, A ;save column info in ACC

MOV A, P1 ;read back Port 0

ANL A, #0F0H ;isolate row lines

CJNE A, #0F0H,KEY\_HIT ;row line active?

MOV A, R7 ;no: move to next

RL A ;column line

DJNZ R6, TEST

CLR C ;no key pressed

RET ;return with C = 0

KEY\_HIT: MOV R7, A ;save scan code in R7

MOV A, #4 ;prepare to calculate

CLR C ;column weighting

SUBB A, R6 ;4 -R6 = column number 0-3

MOV R6, A ;save in R6

MOV A, R7 ;restore scan code

SWAP A ;put in low nibble

MOV R5, #4 ;use R5 as counter

AGAIN: RRC A ;rotate for row num until 0

JNC DONE ;done when C = 0,

INC R6 ;add 4 to keycode to go to next

INC R6 ;row until active row found

INC R6

INC R6

DJNZ R5, AGAIN

DONE: SETB C ;C = 1 (key pressed)

MOV A, R6 ;hex code in A (whew! ! !)

RET

OUTCHR: JB A.3, H8\_f

JB A.2, H4\_7

JB A.1, H2\_3

JB A.0, H1

MOV P2, #03H ;0

RET

H1: MOV P2, #9FH ;1

RET

H2\_3: JB A.0, H3

MOV P2, #25H ;2

RET

H3: MOV P2, #0DH ;3

RET

H4\_7: JB A.1, H6\_7

JB A.0, H5

MOV P2, #99H ;4

RET

H5: MOV P2, #49H ;5

RET

H6\_7: JB A.0, H7

MOV P2, #41H ;6

RET

H7: MOV P2, #1FH ;7

RET

H8\_f: JB A.2, Hc\_f

JB A.1, Ha\_b

JB A.0, H9

MOV P2, #01H ;8

RET

H9: MOV P2, #09H ;9

RET

Ha\_b: JB A.0, Hb

MOV P2, #11H ;A

RET

Hb: MOV P2, #0C1H ;B

RET

Hc\_f: JB P3.5, He\_f

JB P3.4, Hd

MOV P2, #63H ;C

RET

Hd: MOV P2, #85H ;D

RET

He\_f: JB A.0, Hf

MOV P2, #61H ;E

RET

Hf: MOV P2, #71H ;F

RET

END

*2. (30%)*

*MCU8051IDE中有提供一個4個數字顯示的7段顯示器的Virtual Hardware，假設我們所要顯示的數字是存放在內部資料記憶體30H ,31H(2個bytes共4個hex數字)，請在MCU8051IDE環境下設計一個程式來不斷的顯示這4個數字在7段顯示器的Virtual Hardware上。你最少必須提供流程圖(或pseudo code)，解釋你的做法，你的程式碼，Virtual Hardware儲存檔。可以讓助教驗證你的程式。*

MAIN()

BEGIN

[set t0 mode 1]

[enable t0 interrupt]

[turn on led3]

[force t0 interrupt to initialize]

WHILE[1] DO

[nothing]

END

T0ISR()

BEGIN

[stop t0]

[reset t0 to -12500]

[start t0]

CASE [p1] OF

11101110: [a = 30H]

11011101: BEGIN

[a = 30H]

[swap a]

END

11101110: [a = 31H]

11011101: BEGIN

[a = 31H]

[swap a]

END

ENDCASE

[call subroutine outchr]

[rotate turned on led]

RETURN()

END

OUTCHR()

BEGIN

CASE [A] OF

00H: BEGIN

[P2 = 03H]

RETURN()

END

01H BEGIN

[P2 = 9FH]

RETURN()

END

02H: BEGIN

[P2 = 25H]

RETURN()

END

03H: BEGIN

[P2 = 0DH]

RETURN()

END

04H: BEGIN

[P2 = 99H]

RETURN()

END

05H: BEGIN

[P2 = 49H]

RETURN()

END

06H: BEGIN

[P2 = 41H]

RETURN()

END

07H: BEGIN

[P2 = 1FH]

RETURN()

END

08H: BEGIN

[P2 = 01H]

RETURN()

END

09H: BEGIN

[P2 = 09H]

RETURN()

END

0AH BEGIN

[P2 = 11H]

RETURN()

END:

0BH BEGIN

[P2 = 0C1H]

RETURN()

END

0CH: BEGIN

[P2 = 63H]

RETURN()

END

0DH BEGIN

[P2 = 85H]

RETURN()

END:

0EH: BEGIN

[P2 = 61H]

RETURN()

END

0FH: BEGIN

[P2 = 71H]

RETURN()

END

ENDCASE

END

ORG 0000H

LJMP Main

ORG 000BH

LJMP T0ISR

ORG 0030H

Main: MOV TMOD, #01H ;T0 MODE 1

MOV IE, #82H

MOV P1, #77H ;01110111

SETB TF0

SJMP $ ;DO NOTHING

T0ISR: CLR TR0

MOV TH0, #0CFH ;12500us DELAY

MOV TL0, #2CH

SETB TR0

JB P1.0, SKIP1

MOV A, 30H

LJMP EXIT ;1110

SKIP1: JB P1.1, SKIP2

MOV A, 30H

SWAP A

LJMP EXIT ;1101

SKIP2: JB P1.2, SKIP3

MOV A, 31H

LJMP EXIT ;1011

SKIP3: MOV A, 31H

SWAP A

LJMP EXIT ;0111

EXIT: LCALL OUTCHR

MOV A, P1

RL A

MOV P1, A

RETI

OUTCHR: JB A.3, H8\_f

JB A.2, H4\_7

JB A.1, H2\_3

JB A.0, H1

MOV P2, #03H ;0

RET

H1: MOV P2, #9FH ;1

RET

H2\_3: JB A.0, H3

MOV P2, #25H ;2

RET

H3: MOV P2, #0DH ;3

RET

H4\_7: JB A.1, H6\_7

JB A.0, H5

MOV P2, #99H ;4

RET

H5: MOV P2, #49H ;5

RET

H6\_7: JB A.0, H7

MOV P2, #41H ;6

RET

H7: MOV P2, #1FH ;7

RET

H8\_f: JB A.2, Hc\_f

JB A.1, Ha\_b

JB A.0, H9

MOV P2, #01H ;8

RET

H9: MOV P2, #09H ;9

RET

Ha\_b: JB A.0, Hb

MOV P2, #11H ;A

RET

Hb: MOV P2, #0C1H ;B

RET

Hc\_f: JB P3.5, He\_f

JB P3.4, Hd

MOV P2, #63H ;C

RET

Hd: MOV P2, #85H ;D

RET

He\_f: JB A.0, Hf

MOV P2, #61H ;E

RET

Hf: MOV P2, #71H ;F

RET

*3. (40%)*

*MCU8051IDE中有提供一個DS1620的Virtual Hardware，他是一顆溫度偵測與控制用的IC，它透過3條線(DQ, CLK, RST)與microcontroller連接。請上網找該IC的datasheet，了解它的功能以及如何使用它。假定我們打算做加熱器控制如下圖所示，如果溫度低於17度C就打開加熱器加熱，溫度如果高於23度C就把加熱器關閉，停止加熱。同時，必須持續的從DS1620Virtual Hardware讀入目前的溫度值存到內部資料記憶體，它可以用MCU8051IDE直接觀察。請設計一個程式利用MCU8051IDE的 DS1620 Virtual Hardware來完成這項工作。你最少必須提供流程圖(或pseudo code)，解釋你的做法，你的程式碼，Virtual Hardware儲存檔。可以讓助教驗證你的程式。*

**P1.0**

**P1.1**

**P1.2**

**P1.3**

**P1.4**

**P1.5**

**8031**

**DQ**

**CLK**

**RST**

**THI**

**TLO**

**TCOM**

**DS1620**

**P1.6**

**Furnace on**

MAIN()

BEGIN

[write config]

[cpu = 1, 1-shot = 0]

[write th = 23]

[write tl = 17]

[start temperature conversion]

WHILE [1] DO BEIGN

[read temperature]

IF [t high = 1] THEN

[turn off furnace]

ELSE IF [t low = 1] THEN

[turn on furnace]

END

END

SEND()

BEGIN

[set counter = 8]

WHILE [counter > 0] BEGIN

[clear clock]

[send one bit]

[set clock]

[counter --]

END

RETURN()

END

READ()

BEGIN

[set counter = 8]

WHILE [counter > 0] BEGIN

[clear clock]

[read one bit]

[set clock]

[counter --]

END

RETURN()

END

DQ EQU P1.0

CLK EQU P1.1

RST EQU P1.2

THI EQU P1.3

TLO EQU P1.4

TCOM EQU P1.5

FURN EQU P1.6

ORG 0000H

CLR FURN

CONF: SETB RST ;INITIATE TRANSFER

MOV A, #0CH ;WRITE CONFIG

CALL SEND

MOV A, #0AH ;CPU=1, 1-SHOT =0

CALL SEND

CLR RST

SETB RST

MOV A, #01H ;WRITE TH

CALL SEND

MOV A, #46 ;TH = 46\*0.5 = 23 deg. C

CALL SEND

CLR RST

SETB RST

MOV A, #02H ;WRITE TL

CALL SEND

MOV A, #34 ;TH = 34\*0.5 = 17 deg. C

CALL SEND

CLR RST

CONV: SETB RST

MOV A, #0EEH ;start temperature conversion

CALL SEND

CLR RST

SENS: SETB RST

MOV A, #0AAH

CALL SEND

CALL READ

PUSH A

CLR RST

JB THI, OFF

JB T LO, ON

SJMP SENS

OFF: CLR FURN

SJMP SENS

ON: SETB FURN

SJMP SENS

SEND: MOV R0, #08H

NEXT: CLR CLK

RRC A

MOV DQ, C

SETB CLK

DJNZ R0, NEXT

RET

READ: MOV R0, #08H

NEXT1: CLR CLK

MOV C, DQ

RRC A

SETB CLK

DJNZ R0, NEXT1

RRC A

RET

END